BIANCA COGO BARBOSA

BRUNA CAROLINE GONÇALVES FERNANDES

RICARDO ALVIM

CRIPTOGRAFIA SIMÉTRICA E ASSIMÉTRICA

Cornélio Procópio

2016

Índice

1- Descrição do ambiente de teste:

A configuração do hardware:

- CPU: Core i5 3320M @ 2,6 Ghz (2 núcleos físicos e 2 núcleos lógicos).
- Memória: 12 Gigabytes DDR 3 @ 1600 Mhz (4 GB + 8 GB).
- HD: Western Disk 1 TB 5400 RPM Sata III.

A memória possuía 197 gigabytes de espaço livre, o computador possuía processador Core i5 3320M, a velocidade de leitura do disco 150 MB/s e foi usado o Windows 10 Pro Insider Preview como sistema operacional.

2- Bibliotecas:

Há várias bibliotecas utilizadas para a implementação de criptografia na linguagem Java, as comuns são: Java Cryptography Extension (JCE), Java Secure Sockets Extension (JSSE), Java Authentication and Authorization Service (JAAS).

A JCE contém as implementações de algoritmos criptográficos: Cifras simétricas (chave secreta), Cifras assimétricas (chave pública e privada), Resumos. Para utiliza-la deve-se importar os pacotes: java.security e javax.crypto. Suporta vários algoritmos através de Providers. Na linha de código, todos os objetos JCE são criados com: getInstance("algorithm name", "provider name"), permitindo à aplicação "pedir" uma implementação de um algoritmo. Os principais objetos desta biblioteca são: Cipher (métodos: Init, Update, DoFinal), Key, KeyGenerator, KeyPair, PublicKey, PrivateKey, KeyPairGenerator, MessageDigest. Além destes existem outros objetos que realizam operações compostas: Signature, SignedObject, SealedObject, CipherStream.

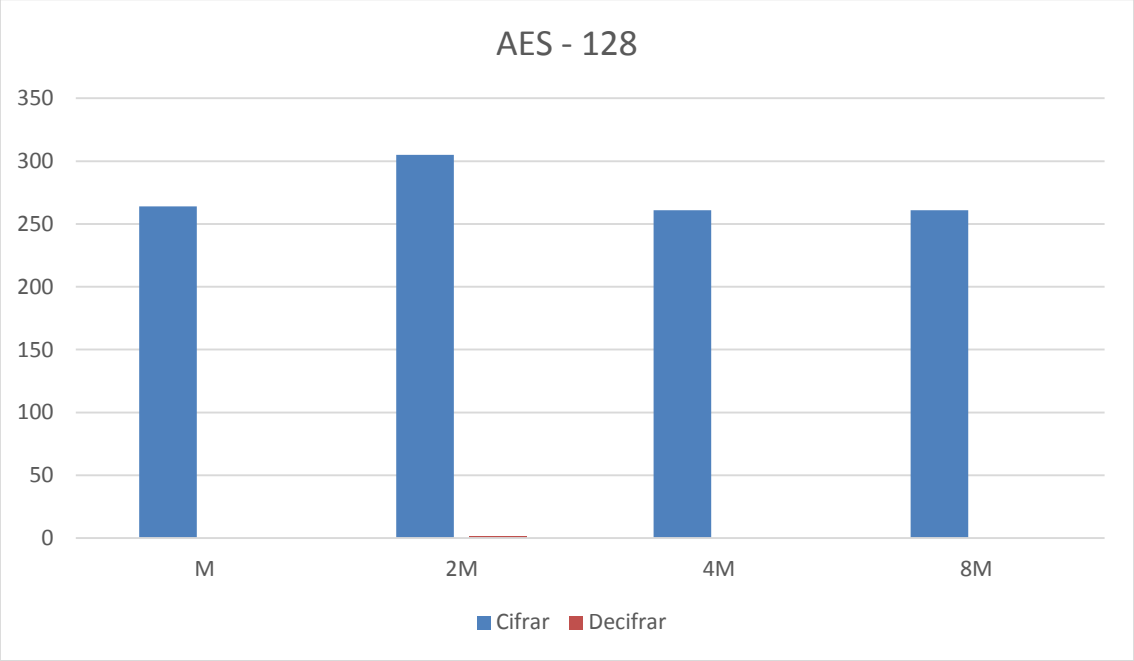
A JSSE é uma biblioteca de sockets que abstraem a utilização de criptografia na comunicação para garantir integridade, cifra e autenticação. É utilizada sobretudo para comunicação na Internet. É uma forma simples de criar canais seguros entre cliente e servidor, com autenticação do servidor, sendo a autenticação do cliente é opcional. Para utiliza-la deve-se importar os pacotes javax.net.ssl, javax.security.net.

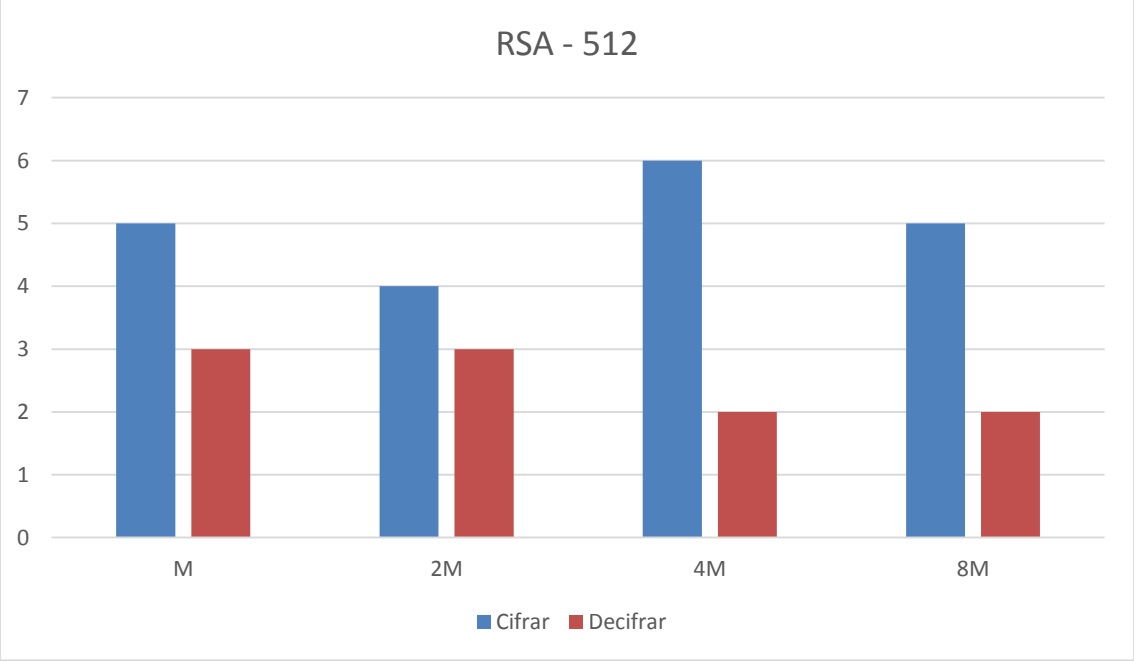
A JAAS tem uma arquitectura Pluggable para autenticação, independente de tecnologias específicas. Seu objetivo é de permitir o controle de acessos dos utilizadores a recursos (autorização) e ter segurança centrada no utilizador (user-centric). Para utilizar este pacote deve-se importar: javax.security.auth.

3- Tabela:

Algoritimo	AES		RSA			
Chave	128		512		1024	
Mensagem	Cifrar	Decifrar	Cifrar	Decifrar	Cifrar	Decifrar
m	264 ms	0 ms	5 ms	3 ms	5 ms	3 ms
2m	305 ms	1 ms	4 ms	3 ms	5 ms	5 ms
4m	261 ms	0 ms	6 ms	2 ms	5 ms	3 ms
8m	261 ms	0 ms	5 ms	2 ms	5 ms	3 ms

4- Gráficos dos resultados:





5- Procedimentos:

Os procedimentos para a implementação foram primeiramente buscar as bibliotecas necessárias para realizar o projeto, após pesquisas, a equipe decidiu que a melhor opção foi a utilização da biblioteca Java Cryptography Extension (JCE), visto que atendia todas as necessidades do projeto.

Após a escolha, foi feita a implementação do projeto, através de pesquisas e fontes de códigos na internet para base, foi implementado o código final. A programação foi feita na linguagem Java utilizando o Netbeans.

6- Avaliação de desempenho:

Observa-se que o algoritmo AES é mais rápido tanto para encriptar quanto para decriptar a mensagem, devido ao uso de uma única chave (criptografia simétrica), em relação ao RSA que utiliza-se um par de chaves (criptografia assimétrica).

Nota-se que no AES conforme aumenta o tamanho da mensagem também aumenta o tempo de cifrar, porem o tempo de decriptação parece não sofrer alteração.

7- Conclusão:

Conclui-se que o algoritmo AES pode ser mais rápido, porém não foi possível implementa-lo com 256 bits, entretanto não possui toda a segurança que o algoritmo RSA possui pelo fato de ser assimétrico e usar 2 chaves, o que torna o RSA mais seguro apesar de seu tempo de decifrar ser um pouco maior.

8- Referência:

Disponível em: <http://disciplinas.ist.utl.pt/leic-sod/2009-2010/labs/08-reqs-n-funcs/1-sec/seguranca-java/index.html>

Disponível em: <https://tics.taxi/criptografia-simetrica-em-java/>

Disponível em: <https://regispires.wordpress.com/2010/11/10/encriptacao-decriptacao-simples-em-java/>